

AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

- 1 1. (Currently amended) A method for dynamically adjusting the
2 aggressiveness of an execute-ahead processor, comprising:
3 executing instructions in an execute-ahead mode, wherein instructions that
4 cannot be executed because of an unresolved data dependency are deferred in a
5 deferred buffer, and other non-deferred instructions are executed in program
6 order, and wherein if a non-data-dependent stall condition is encountered, the
7 execute-ahead processor enters a scout mode, wherein instructions are
8 speculatively executed to prefetch future loads, but results are not committed to
9 the architectural state of the execute-ahead processor;
10 executing deferred instructions in a deferred mode in response to
11 determining if that an unresolved data dependency is~~was~~ resolved during the
12 execute-ahead mode;
13 ~~executing deferred instructions in a deferred mode;~~
14 ~~wherein if some instructions are deferred again during the deferred mode,~~
15 ~~the method further comprises,~~
16 waiting for the deferred buffer to empty in response to determining
17 ~~whether that~~ an amount of work accomplished during the execute-ahead mode
18 exceeds a predetermined threshold; and;
19 if so,
20 ~~waiting for the deferred buffer to empty, and~~
21 returning to a normal execution mode;

22 | otherwise resuming execution in ~~execute-ahead mode~~.

1 | 2. (Canceled).

1 | 3. (Currently amended) The method of claim 1, further comprising
2 | ~~wherein resuming execution in the non-aggressive mode involves resuming~~
3 | ~~execution~~ executing instructions in a non-aggressive execute-ahead mode,
4 | wherein if a non-data-dependent stall condition is encountered, the execute-ahead
5 | processor does not enter the scout mode, but instead waits for the non-data-
6 | dependent stall condition to be resolved, or for an unresolved data dependency to
7 | return, before proceeding.

1 | 4. (Currently amended) The method of claim 1, wherein prior to executing
2 | instructions in the execute-ahead mode, the method further comprises entering the
3 | execute-ahead mode by:
4 | issuing instructions for execution in program order during ~~a~~ the normal
5 | execution mode;
6 | upon encountering ~~an~~ a first unresolved data dependency during execution
7 | of an instruction,
8 | generating a checkpoint that can subsequently be used to
9 | return execution to the point of the instruction, and
10 | executing subsequent instructions in the execute-ahead
11 | mode.

1 | 5. (Currently amended) The method of claim 4, wherein if the first
2 | unresolved data dependency is finally resolved, the method further comprises
3 | using the checkpoint to resume execution in the normal execution mode, ~~from the~~

4 | ~~launch point instruction (the instruction that originally encountered the launch~~
5 | ~~point stall condition).~~

1 6. (Original) The method of claim 1, wherein executing deferred
2 instructions in the deferred mode involves:
3 issuing deferred instructions for execution in program order;
4 deferring execution of deferred instructions that still cannot be executed
5 because of unresolved data dependencies; and
6 executing other deferred instructions that are able to be executed in
7 program order.

1 7. (Currently amended) The method of claim 6, wherein if all deferred
2 instructions are executed in the deferred mode, the method further comprises
3 | returning to a the normal execution mode to resume normal program execution
4 | from the point where the execute-ahead mode left off.

1 8. (Currently amended) The method of claim 1, wherein the unresolved
2 | data dependency ~~can include~~ is one of:
3 a use of an operand that has not returned from a preceding load miss;
4 a use of an operand that has not returned from a preceding translation
5 lookaside buffer (TLB) miss;
6 a use of an operand that has not returned from a preceding full or partial
7 read-after-write (RAW) from store buffer operation; and
8 a use of an operand that depends on another operand that is subject to an
9 unresolved data dependency.

1 9. (Currently amended) The method of claim 1, wherein the non-data-
2 | dependent stall condition ~~can include~~ is one of:

3 a memory barrier operation;
4 a load buffer full condition; and
5 a store buffer full condition.

1 10. (Currently amended) An apparatus that dynamically adjusts the
2 aggressiveness of an execute-ahead processor, comprising:
3 an execution mechanism configured to:
4 execute instructions in an execute-ahead mode, wherein
5 instructions that cannot be executed because of an unresolved data
6 dependency are deferred in a deferred buffer, and other non-
7 deferred instructions are executed in program order, and wherein if
8 a non-data-dependent stall condition is encountered, the execution
9 mechanism is configured to enter a scout mode, wherein
10 instructions are speculatively executed to prefetch future loads, but
11 results are not committed to the architectural state of the execute-
12 ahead processor;
13 execute deferred instructions in a deferred mode in
14 response to determining wherein if that an unresolved data
15 dependency is was resolved during the execute-ahead mode; ~~the~~
16 ~~execution mechanism is configured to execute deferred instructions~~
17 ~~in a deferred mode;~~
18 wait for the deferred buffer to empty in response to
19 determining that an amount of work accomplished during the
20 execute-ahead mode exceeds a predetermined threshold; and to
21 return to a normal execution mode.
22 ~~wherein if some instructions are deferred again during the deferred mode,~~
23 ~~the execution mechanism is configured to;~~

24 | ~~determine whether an amount of work accomplished during~~
25 | ~~execute-ahead mode exceeds a predetermined threshold,~~
26 | ~~if so,~~
27 | ~~waiting for the deferred buffer to empty, and~~
28 | ~~returning to normal execution mode,~~
29 | ~~otherwise to resume execution in execute-ahead mode.~~

1 | 11. (Canceled).

1 | 12. (Currently amended) The apparatus of claim 10, wherein ~~while~~
2 | ~~resuming execution in the non-aggressive execution mode,~~ the execution
3 | mechanism is configured to execute instructions ~~resume execution~~ in a non-
4 | aggressive execute-ahead mode, wherein if a non-data-dependent stall condition is
5 | encountered, the execution mechanism does not enter the scout mode, but instead
6 | waits for the non-data-dependent stall condition to be resolved, or for an
7 | unresolved data dependency to return, before proceeding.

1 | 13. (Currently amended) The apparatus of claim 10, wherein prior to
2 | executing instructions in the execute-ahead mode, the execution mechanism is
3 | configured to enter the execute-ahead mode by:
4 | issuing instructions for execution in program order during ~~a~~ the normal
5 | execution mode;
6 | upon encountering ~~an~~ a first unresolved data dependency during execution
7 | of an instruction,
8 | generating a checkpoint that can subsequently be used to
9 | return execution at to the point of the instruction, and
10 | executing subsequent instructions in the execute-ahead
11 | mode.

1 14. (Currently amended) The apparatus of claim 13, wherein if the first
2 unresolved data dependency is finally resolved, the execution mechanism is
3 configured to use the checkpoint to resume execution in the normal execution
4 mode, ~~from the launch point instruction (the instruction that originally~~
5 ~~encountered the launch point stall condition).~~

1 15. (Original) The apparatus of claim 10, wherein while executing
2 deferred instructions in the deferred mode, the execution mechanism is configured
3 to:
4 issue deferred instructions for execution in program order;
5 defer execution of deferred instructions that still cannot be executed
6 because of unresolved data dependencies; and to
7 execute other deferred instructions that are able to be executed in program
8 order.

1 16. (Currently amended) The apparatus of claim 15, wherein if all deferred
2 instructions are executed in the deferred mode, the execution mechanism is
3 configured to return to a the normal execution mode to resume normal program
4 execution from the point where the execute-ahead mode left off.

1 17. (Currently amended) The apparatus of claim 10, wherein the
2 unresolved data dependency ~~can include~~ is one of:
3 a use of an operand that has not returned from a preceding load miss;
4 a use of an operand that has not returned from a preceding translation
5 lookaside buffer (TLB) miss;
6 a use of an operand that has not returned from a preceding full or partial
7 read-after-write (RAW) from store buffer operation; and

8 a use of an operand that depends on another operand that is subject to an
9 unresolved data dependency.

1 18. (Currently amended) The apparatus of claim 10, wherein the non-data-
2 dependent stall condition ~~can include~~ is one of:
3 a memory barrier operation;
4 a load buffer full condition; and
5 a store buffer full condition.

1 19. (Currently amended) A computer system that dynamically adjusts the
2 aggressiveness of an execute-ahead processor, comprising:
3 an execute-ahead processor;
4 a memory;
5 an execution mechanism within the execute-ahead processor configured
6 to:
7 execute instructions in an execute-ahead mode, wherein
8 instructions that cannot be executed because of an unresolved data
9 dependency are deferred in a deferred buffer, and other non-
10 deferred instructions are executed in program order, and wherein if
11 a non-data-dependent stall condition is encountered, the execution
12 mechanism is configured to enter a scout mode, wherein
13 instructions are speculatively executed to prefetch future loads, but
14 results are not committed to the architectural state of the execute-
15 ahead processor;
16 execute deferred instructions in a deferred mode in
17 response to determining wherein if that an unresolved data
18 dependency ~~is~~ was resolved during the execute-ahead mode; ~~the~~

19 ~~execution mechanism is configured to execute deferred instructions~~
20 ~~in a deferred mode;~~
21 wait for the deferred buffer to empty in response to
22 determining that an amount of work accomplished during the
23 execute-ahead mode exceeds a predetermined threshold; and to
24 return to a normal execution mode.
25 ~~wherein if some instructions are deferred again during the deferred mode,~~
26 ~~the execution mechanism is configured to,~~
27 ~~determine whether an amount of work accomplished during~~
28 ~~execute-ahead mode exceeds a predetermined threshold,~~
29 ~~if so,~~
30 ~~waiting for the deferred buffer to empty, and~~
31 ~~returning to normal execution mode,~~
32 ~~otherwise to resume execution in execute-ahead mode.~~

1 20. (Canceled).

1 21. (Currently amended) The computer system of claim 19, wherein ~~while~~
2 ~~resuming execution in the non-aggressive execution mode,~~ the execution
3 mechanism is configured to execute instructions ~~resume execution~~ in a non-
4 aggressive execute-ahead mode, wherein if a non-data-dependent stall condition is
5 encountered, the execution mechanism does not enter the scout mode, but instead
6 waits for the non-data-dependent stall condition to be resolved, or for an
7 unresolved data dependency to return, before proceeding.